

1. Basic Katalon Studio Knowledge

Q: What is Katalon Studio? How does it differ from Selenium?

A: Katalon Studio is an all-in-one test automation solution for Web, API, Mobile, and Desktop applications. Unlike Selenium, which is a framework requiring programming expertise, Katalon provides a user-friendly interface, built-in keywords, and dual scripting modes (manual and script) to simplify test creation.

Q: What are the main components of Katalon Studio?

A: Key components include:

- Test Cases
- Test Suites
- Object Repository
- Test Data
- Custom Keywords
- Reports

Q: Can you explain the object repository in Katalon?

A: The Object Repository stores UI elements (test objects) that are used in test cases. It promotes reuse and maintenance by centralizing element definitions.

Q: What are test cases and test suites in Katalon Studio?

A: A test case is a single automated test. A test suite groups multiple test cases to run in sequence. You can also create a Test Suite Collection to run multiple suites with different profiles.

Q: What types of testing can you perform with Katalon Studio?

A: Web testing, Mobile testing, API testing, Desktop app testing, and Data-driven testing.

2. Scripting and Test Creation

Q: How do you record a test in Katalon Studio?

A: Use the “Record Web” or “Record Mobile” feature to interact with the application. Katalon captures actions and converts them into steps in a test case.

Q: What are the different ways to create test steps in Katalon?

A: You can add steps manually using the Manual view, or write Groovy/JavaScript code in Script view. Test steps can also be added by drag-and-drop or importing from existing test objects.

Q: How do you use variables and parameters in Katalon test cases?

A: You can define global variables in execution profiles or use test case variables and data-driven inputs through Test Data binding.

Q: Explain the difference between Manual and Script view in Katalon.

A: Manual view allows users to build test steps using a GUI without coding. Script view allows editing the underlying Groovy code for full flexibility.

Q: How do you use custom keywords in Katalon Studio?

A: Custom keywords are user-defined methods stored under the Keywords folder. You define them in Groovy files and call them in test cases using CustomKeywords. syntax.

3. Automation and Execution**Q: How do you handle waits in Katalon Studio?**

A: Katalon supports both implicit and explicit waits. Built-in keywords like `WebUI.waitForElementVisible()` or `WebUI.waitForPageLoad()` are used to synchronize the test with the application state.

Q: How do you manage browser compatibility in test cases?

A: You can configure the browser in the execution settings or use execution profiles. Katalon supports Chrome, Firefox, Edge, Safari, and Remote execution (via Selenium Grid or BrowserStack).

Q: Can you schedule test execution in Katalon? How?

A: Yes. You can use Katalon TestOps or integrate with CI tools like Jenkins or Azure DevOps to schedule and run automated tests.

Q: How do you integrate Katalon Studio with CI/CD tools like Jenkins?

A: Use the Katalon command-line interface (CLI) in Jenkins pipelines to execute test suites. TestOps can also collect and report the results after execution.

Q: What are execution profiles in Katalon Studio?

A: Execution profiles are sets of global variables (like URLs, credentials, etc.) that let you run the same test cases in different environments (Dev, QA, Prod).

4. Object Spy and Object Handling**Q: How do you capture and use test objects in Katalon Studio?**

A: Use the Spy Web or Spy Mobile feature to inspect the UI and capture elements. Katalon stores them in the Object Repository for use in test steps.

Q: What is Smart XPath? How does it help in object identification?

A: Smart XPath automatically generates the most stable XPath by analyzing the DOM. It improves object recognition when the UI structure changes slightly.

Q: How do you deal with dynamic objects in Katalon Studio?

A: Use parameterized test objects. You can define dynamic XPaths with placeholders (e.g., `${itemName}`) and replace them at runtime using `findTestObject()` with `TestObjectProperty`.

5. Reporting and Logs**Q: What kind of reports does Katalon generate?**

A: Katalon generates HTML, CSV, PDF, and JUnit-compatible reports with detailed logs, screenshots, and execution statuses.

Q: How do you analyze test results in Katalon Studio?

A: Use the Log Viewer to inspect step-by-step execution details. Failed steps show stack traces and screenshots for debugging.

Q: How do you export test reports?

A: Reports are automatically saved in the Reports folder. You can export them manually or integrate with Katalon TestOps for cloud-based reporting.

6. Integration and Plugins

Q: What tools can Katalon Studio integrate with?

A: Katalon supports integration with:

- Jenkins, Azure DevOps, Git
- Jira (for defect tracking)
- Slack and Email (for alerts)
- TestOps and Docker

Q: Have you used Katalon TestOps? If so, explain its benefits.

A: Yes. TestOps provides centralized test management, analytics, scheduling, and team collaboration features. It also helps visualize trends and coverage.

Q: Have you used any Katalon plugins? Which ones and why?

A: Yes, for example:

- Basic Auth Plugin for authentication handling
 - PDF Keyword plugin to validate PDF content
 - Jira Integration plugin to link test cases to issues
-

7. API Testing (If applicable)

Q: How do you perform API testing in Katalon Studio?

A: You can create API test objects (REST/SOAP) using the "Web Service Request" type. Katalon allows setting methods, headers, body content, and verification scripts using built-in keywords like `WS.sendRequest()` and `WS.verifyResponseStatusCode()`.

Q: How do you validate JSON or XML responses?

A: Use assertions in the Verification tab or in scripts with built-in methods like:

- `WS.verifyElementPropertyValue(response, 'key', 'expectedValue')`
- Use Groovy's `JSONSlurper` or `XmlSlurper` for complex parsing and assertions.

Q: What types of assertions are available in Katalon for API testing?

A: You can verify:

- Status code (e.g., 200, 404)
- Response time
- Specific values in the response body
- Header values

- Response schema (via scripts)
-

8. Real-world Scenarios

Q: Describe a challenging bug you encountered using Katalon and how you resolved it.

A: One challenge was handling dynamic web elements that changed IDs on every page load. I resolved it by using parameterized XPaths and Smart XPath, and stored identifiers in global variables for dynamic use.

Q: How do you handle test data in Katalon?

A: Katalon supports:

- Internal test data (Excel, CSV, Database)
- Test Data binding to test cases or test suites
- Execution profiles for environment-specific data

Q: How would you structure a large test automation project in Katalon?

A: Use modular test cases with reusable components, organized folders for test cases, test data, and object repository. Implement custom keywords and use Git for version control. Test Suites and Test Suite Collections handle different test flows.

9. Best Practices and Troubleshooting

Q: What are some best practices for writing maintainable Katalon tests?

A:

- Use descriptive names for test cases and objects
- Centralize object handling via Object Repository
- Use custom keywords for repetitive logic
- Parameterize data and URLs via execution profiles
- Use Git for version control and team collaboration

Q: How do you handle flaky tests in Katalon?

A:

- Add waits and retry logic where necessary
- Use Smart XPath or relative locators
- Stabilize test data and environment
- Log and review failure patterns regularly

Q: What steps would you take if Katalon Studio crashes or becomes unresponsive?

A:

- Check memory usage and increase JVM heap size in the .ini file

- Clear the workspace cache
 - Ensure plugins and drivers are updated
 - Review Katalon logs in the .log file under the Katalon workspace folder
-

10. Behavioral / Soft Skills

Q: How do you prioritize test cases for automation?

A: Based on risk, repeatability, stability, and business criticality. Smoke and regression scenarios usually come first, followed by frequently used user paths and data-driven flows.

Q: How do you collaborate with developers and other QA team members?

A: Through daily stand-ups, Jira for task tracking, code reviews for shared scripts, and Slack/MS Teams for communication. Shared documentation and Git repositories help streamline collaboration.

Q: How do you stay updated with new features or best practices in test automation?

A: I follow Katalon blogs, forums, webinars, and GitHub repos. I also engage in QA communities like Stack Overflow and regularly explore automation trends and tools like Playwright and Cypress for benchmarking.